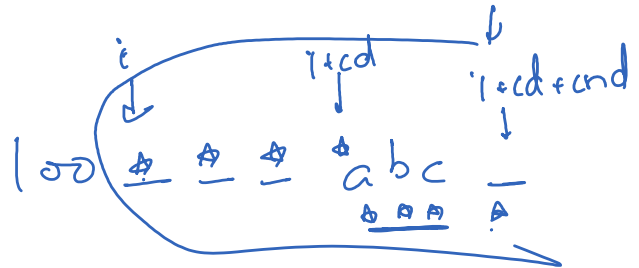
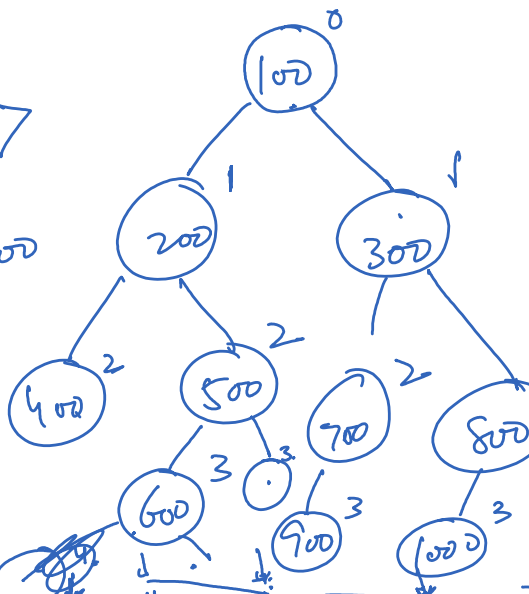


10 - 20 - 40 - 50 - 70 - 80 - 100 - 120 - 140 - 160 - 180 - 200 - 220 - 240 - 260 - 280 - 300 - 320 - 340 - 360 - 380 - 400 - 420 - 440 - 460 - 480 - 500 - 520 - 540 - 560 - 580 - 600 - 620 - 640 - 660 - 680 - 700 - 720 - 740 - 760 - 780 - 800 - 820 - 840 - 860 - 880 - 900 - 920 - 940 - 960 - 980 - 1000



d #
nd #
NM < 9, 15 >
0 - 100
1 - 200, 300



```

public TreeNode helper(String str, int depth){
    int d = 0;
    while(i + d < str.length() && str.charAt(i + d) == '-'){
        d++;
    }
    if(d != depth){
        return null;
    }

    int nd = 0;
    while(i + d + nd < str.length() && str.charAt(i + d + nd) != '-'){
        nd++;
    }

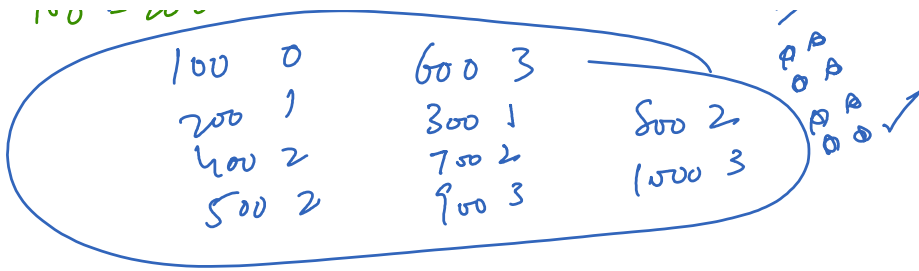
    int val = Integer.parseInt(str.substring(i + d, i + d + nd));
    i = i + d + nd;

    TreeNode node = new TreeNode(val);
    node.left = helper(str, depth + 1);
    node.right = helper(str, depth + 1);

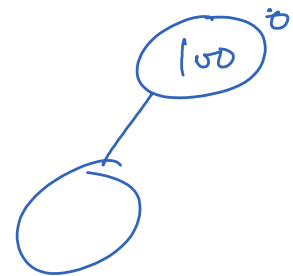
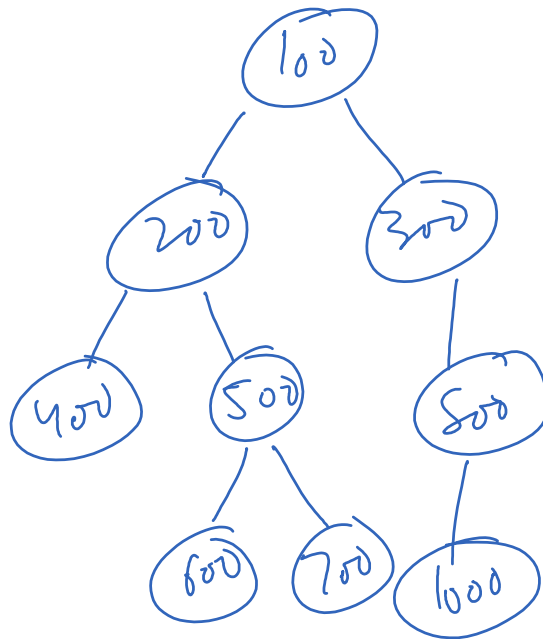
    return node;
}
  
```

C → s.l ~ no of nodes

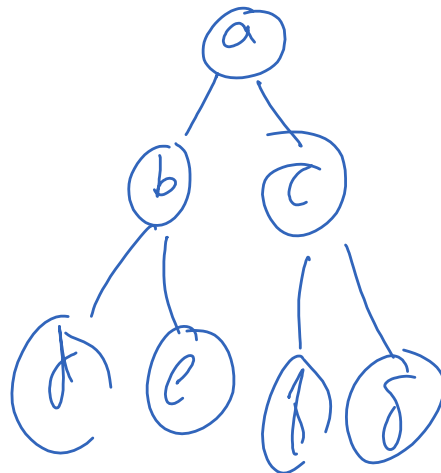




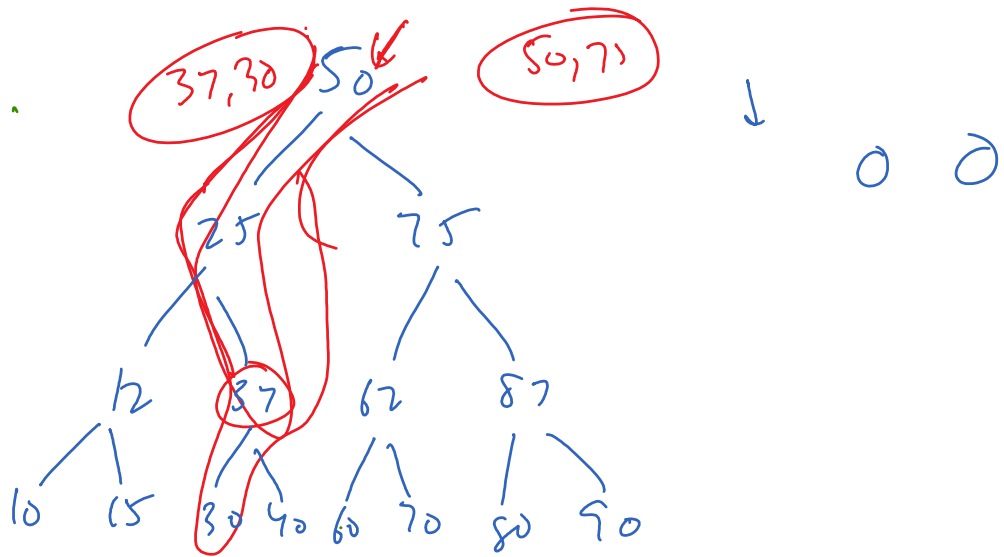
2n



↓
 100 - 200 - 400 - 500 - 600 - 700 - 800 - 1000



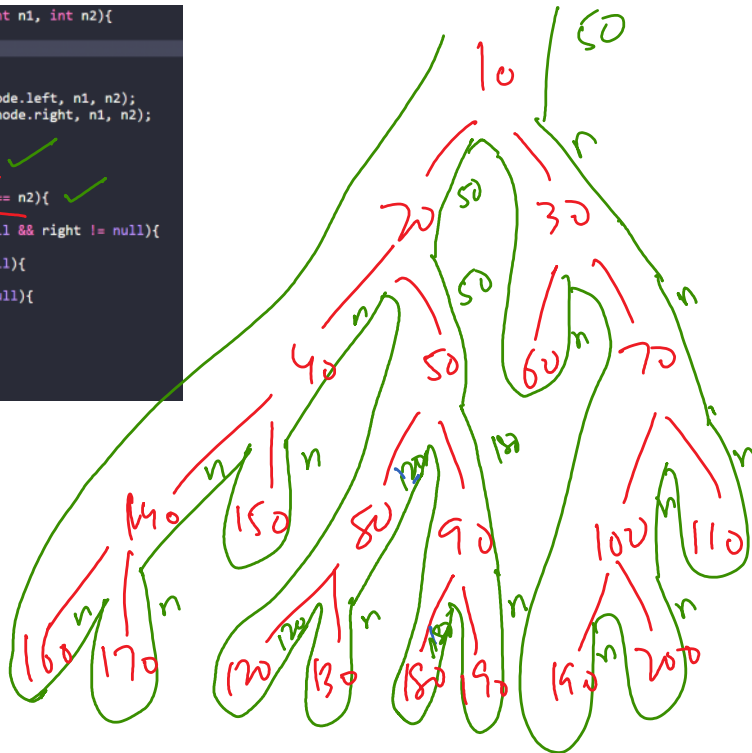
$O(s.l)$



```
Node helper(Node node, int n1, int n2){
    if(node == null){
        return null;
    }

    Node left = helper(node.left, n1, n2);
    Node right = helper(node.right, n1, n2);

    if(node.data == n1){
        return node;
    } else if(node.data == n2){
        return node;
    } else if(left != null && right != null){
        return node;
    } else if(left != null){
        return left;
    } else if(right != null){
        return right;
    } else {
        return null;
    }
}
```



LCA in a BT

11:14 to 11:24

50, 100

